

10-10-00

A

10/06/00
JC929 U.S. PTO

Patent
Attorney's Docket No. 001580-713

"Express Mail" mailing label No. EL 521771476 US
Date of Deposit 6 October, 2000

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to BOX PATENT APPLICATION, Commissioner for Patents, Washington, D.C. 20231.

SEINI MATANGI
(Typed or printed name of person mailing paper or fee)

(Signature of person mailing paper or fee)

JC914 U.S. PTO
09/680990
10/06/00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

UTILITY PATENT
APPLICATION TRANSMITTAL LETTER

Box PATENT APPLICATION
Commissioner for Patents
Washington, D.C. 20231

Sir:

Enclosed for filing is the utility patent application of Rusty Tucker for Improved Connectionless Protocol.

Also enclosed are:

- ☒ Three sheet(s) of [] formal ☒ informal drawing(s);
- [] a claim for foreign priority under 35 U.S.C. §§ 119 and/or 365 is [] hereby made to _
filed in _ on _;
[] in the declaration;
- [] a certified copy of the priority document;
- [] a General Authorization for Petitions for Extensions of Time and Payment of Fees;
- [] _____ statement(s) claiming small entity status;
- [] an Assignment document;
- [] an Information Disclosure Statement; and
- [] Other: _____



21839

09680990 100600

☒ An ☒ executed ☐ unexecuted declaration of the inventor(s)
☐ also is enclosed ☒ will follow.

☐ Please amend the specification by inserting before the first line the sentence --This application claims priority under 35 U.S.C. §§119 and/or 365 to filed in on ; the entire content of which is hereby incorporated by reference.--

☐ A bibliographic data entry sheet is enclosed.

☒ The filing fee has been calculated as follows ☐ and in accordance with the enclosed preliminary amendment:

C L A I M S					
	NO. OF CLAIMS		EXTRA CLAIMS	RATE	FEE
Basic Application Fee					\$690.00
Total Claims	80	MINUS 20 =	60	x \$18.00	\$ 1,080.00
Independent Claims	8	MINUS 3 =	5	x \$78.00	\$ 390.00
If multiple dependent claims are presented, add \$260.00 (104)					0
Total Application Fee					\$ 2,160.00
If verified Statement claiming small entity status is enclosed, subtract 50% of Total Application Fee					0
Add Assignment Recording Fee \$40.00 (581) if Assignment document is enclosed					0
TOTAL APPLICATION FEE DUE					\$ 2,160.00

☒ This application is being filed without a filing fee. Issuance of a Notice to File Missing Parts of Application is respectfully requested.

☐ A check in the amount of \$ is enclosed for the fee due.

☐ Charge \$ to Deposit Account No. 02-4800 for the fee due.

☒ The Commissioner is hereby authorized to charge any appropriate fees under 37 C.F.R. §§ 1.16, 1.17 and 1.21 that may be required by this paper, and to credit any overpayment, to Deposit Account No. 02-4800.


Please address all correspondence concerning the present application to:

James W. Peteson
Burns, Doane, Swecker & Mathis, L.L.P.
P.O. Box 1404
Alexandria, Virginia 22313-1404.

Respectfully submitted,

BURNS, DOANE, SWECKER & MATHIS, L.L.P.

Date: October 6, 2000

By: 
Anthony T. Cascio
Registration No. 26,904

P.O. Box 1404
Alexandria, Virginia 22313-1404
(650) 622-2300

00900T" 06608560

Improved Connectionless Protocol

Field of Invention

The present invention relates generally to protocols for the transmission of packets over a computer network, and more particularly, to an improved connectionless protocol.

Background of the Invention

In a computer network in which a plurality of clients communicate with one or more servers, a protocol is necessary to establish the rules by which the information is transmitted to effect such communication. Generally, the information is sent from the server in successive packets of information, with each of the packets transversing their own paths through the network, until such time as the packets are reassembled at the destination client. To establish the communication, each packet contains, in addition to a data segment of the total information to be sent, the source address of the server in the network and the destination address of the client in the network.

A commonly used protocol used to establish these connections is the Transmission Control Protocol ("TCP"). TCP is used along with the Internet Protocol ("IP") to send data in the form of message units between computers over the Internet. While IP takes care of handling the actual delivery of the data, TCP takes care of keeping track of the packets that a message is divided into for efficient routing through the Internet.

For example, when an HTML file is sent from a web server, the TCP program layer in that server divides the file into one or more packets, numbers the packets, and then forwards them individually to the IP program layer. Although each packet has the same destination IP address, each packet may get routed differently through the network. At the client, TCP reassembles the individual packets and waits until they have arrived to display the re-assembled HTML file.

TCP is also known as a connection-oriented protocol, which means that a connection is established and maintained until such time as the message or messages to be exchanged by the application programs at each end of the connection have been exchanged. TCP is responsible for ensuring that a message is divided into the packets that IP manages and for reassembling the packets back into the complete message at the other end.

Another protocol to transmit information is the User Datagram Protocol ("UDP"). UDP is a communications method that offers a limited amount of service when messages are exchanged between computers in a network that uses IP. UDP is an alternative to TCP and, together with IP, is sometimes referred to as UDP/IP. Like TCP, UDP uses the IP to actually get a data unit, or datagram, from one computer to another.

Unlike TCP, however, UDP does not provide the service of dividing a message into packets or datagrams, and reassembling them at the other end. Specifically, UDP does not provide sequencing of the packets that the data arrives in. Accordingly, the application program that uses UDP must be able to ascertain that the entire message has arrived and is in the right order. Network applications that want to save processing time because they have very small data units to exchange (and therefore very little message reassembling to do) may usually use UDP instead of

TCP. UDP does provide two services not provided by the IP layer. It provides a port number to help distinguish different user requests and, optionally, a checksum capability to verify that the data arrived intact.

5 For streaming applications over a computer network, both UDP and TCP have significant disadvantages and limitations. For example, since TCP requires that all packets are received prior to reassembly at the client, any missing packets could cause the streaming application running at the client to be paused due to the lack of data. UDP also requires that all of the packets are received and also received in the right order, since there is no sequencing of the packets. For a large number of packets in
10 a streaming application, the packets may arrive out of order, resulting in degradation of the output of the streaming application.

Other features of TCP further disadvantageously limit the ability to deliver packets for streaming applications in a timely manner. For example, TCP requires that each packet sent in the stream must be received and each packet delivered to the
15 client in the order sent. This requirement of TCP is especially limiting and disadvantageous for a streaming application. If the server sends four sequential packets, and packet 1 is lost, the client running the streaming application cannot receive the remaining three packets from the TCP protocol stack until it receives the first packet. If the client receives the first packet sufficiently late, all the previously
20 received packets may be useless due to the tardy reception. In any event, the output of the streaming application will be degraded or possibly halted.

There is therefore needed a connectionless protocol which overcomes one or more disadvantages and limitations of the prior art herein above enumerated.

Summary of the Invention

It is an object of the present invention to overcome one or more disadvantages and limitations of the prior art herein above enumerated.

5 It is an object of the present invention to provide a connectionless protocol which does not require receipt of all packets by the client, nor require receipt of packets in order of transmission.

10 According to one aspect of the present invention, a method for real time transmission of information content between a network server and a network client comprises the steps of transmitting successive packets of the content from the server to a retransmit module, assigning at the retransmit module to each of the packets a sequence number and a first timer, transmitting further each of the packets from the retransmit module to the network client, transmitting from the network client to the retransmit module an acknowledgment for each of the packets received at the network client, retransmitting from the retransmit module any of the packets upon expiration of the first timer assigned thereto prior to an acknowledgment for the any one of the packets being received, and removing from the retransmit module any of the packets upon an event determination prior to an acknowledgement for such anyone of the packets being received.

20 In a second aspect of the present invention, a method for acknowledging receipt of packets sent from a network server to a network client comprises steps of transmitting successively packets from the server, receiving at the client several of the packets, placing into a coalesced acknowledgment an ID of a first one of the several of the packets received at the client, adding to the coalesced acknowledgment a bit map

identifying selected other ones of the several of the packets received at the client, and transmitting to the server the coalesced acknowledgment.

Features of the present invention are flow control, congestion avoidance and packet recovery without any necessity for guaranteed or in order delivery. Flow control is achieved by limiting the amount of unacknowledged data to a given window size. When the window is full, further transmission is delayed until packets are acknowledge or expired. A slow start algorithm is also provided to adapt the rate of packet transmission to the client. Congestion is avoided by using the TCP exponential back off. Robust packet recovery is obtained by using Karn's algorithm for round trip estimation. The order of packet deliver is not required nor guaranteed. Acknowledgments are sent on a packet by packet basis. In another embodiment of the invention, acknowledgments may be coalesced.

These and other objects, advantages and features of the present invention will become readily apparent to those skilled in the art from a study of the following Description of the Exemplary Preferred Embodiments when read in conjunction with the attached Drawing and appended Claims.

Brief Description of the Drawing

Figure 1 is a schematic block diagram of a computer network constructed according to the principles of the present invention.

Figure 2 is a table useful to describe the function of the retransmit module of Fig. 1.

Figure 3 is a flow chart useful to describe one method of the present invention.

Figure 4 is a flow chart useful to describe another method of the present invention.

Description of the Exemplary Preferred Embodiments

With reference to Figure 1, there is shown a network 10 including a server 12, a client 14, and a retransmit module 16. The server is adapted to send successive packets, such as packets 18a, 18b, into the network 10. The client 14 is adapted to receive such packets 18a, 18b, from the network 10. As is known, the packets 18a and 18b may traverse differing paths through the Internet 20, which may also be part of the network 10. Each of the server 12 and client 14 respectively has a computer readable medium 12a, 14a in which a program is stored which implements the below described inventive procedures, processes and methods.

In accordance with the present invention, the packets are first sent from the server to the retransmit module 16 as seen at 30, 32 in Fig. 3. The retransmit module 16 assigns a sequence number or ID to each packet, a first timer and second timer as best seen in the table of Figure 2. The first timer establishes a first time duration, or retransmit time, upon the expiration of which the packet 18a, 18b to which it is assigned is retransmitted. The second timer establishes a "time to live" for the packet 18a, 18b, to which it is assigned. Upon expiration of the second timer, the packet 18a, 18b to which it is assigned is removed from the retransmit module 16 even if no acknowledgement for receipt of such packet 18a, 18b has been received.

When the packets 18a, 18b are sent to the retransmit module, they are retained therein, as well as transmitted through the network 10 as seen at 34 in Fig. 3. If an acknowledgment (ACK) is not returned as indicated in 36 to the retransmit module 16 for any of the packets 18a, 18b, prior to the expiration of the first timer associated

therewith, indicated at 38, such packet 18a, 18b will be retransmitted, and the first timer for the retransmit time may also be reset as indicated at 40.

Although it has been described above that, if an acknowledgment is not received for such packet 18a, 18b prior to the expiration of the second timer for the time to live, the packet will be removed from the retransmit module 16, several other methods or procedures may be utilized. For example, the packet 18a, 18b may be removed from the retransmit module upon the occurrence of a predetermined event prior to receipt of any acknowledgement therefor as indicated at 42. Such event may be a preselected number of retransmits for such packet 18a, 18b, or the subsequent transmission of another packet containing more critical information, as discussed herein below. If an acknowledgment is received prior to the expiration of either timer, the packet will also be removed from the retransmit module 16, as indicated at 44.

In one embodiment of the present invention, the first timer and the second timer may be assigned on the basis of the criticality of the information contained in the packet. For example in a streaming video application, certain packets may be designated as frame packets necessary for reconstructing an entire video frame at the client 14. Successive packets following the frame packets may then be designated as differential packets, which contain only information which has changed within the frame. Differential packets may then be given shorter time to live than frame packets which contain more valuable information for the streaming application. By allowing packets to expire, the stream of packets keeps moving forward. Rather than retransmit a stale packet, a fresher and potentially more important packet is transmitted in its place. The expiration of the stale packet could be based either on the time to live associated with the second timer, or alternatively, on the event of a subsequent frame packet being transmitted, for example.

In another aspect of the present invention, the stream of packets 18a, 18b, is regulated, or slowly started, to increase the allowable window size so that competing streams can find equilibrium in their transit rate when they are first starting up, and also when a retransmit occurs. This slow start is effective to minimize the number of lost packets by keeping total bandwidth demand within the bandwidth constraints along the path between the server 12 and the client 14.

While a stream is sending, the window size is constrained to the lesser of a "Congestion Window", or CWND and client window sizes. The client window must be advertised using RTSP during protocol negotiation. The size must be no greater than the size of the UDP socket input buffer at the client 14. CWND is initialized to the maximum segment size (MSS).

For each packet 18a, 18b acknowledged by the client, CWND is increased by the sized of the acknowledged packet 18a, 18b. CWND is continually increased for each acknowledgment until the CWND is greater than or equal to a predetermined slow start threshold (SSTH). Once the SSTH is reached, the CWND is increased by the MSS for each window of ACK's.

CWND may be constrained by the MSS at the minimum and the client window size at the maximum. SSTH is maintained by an exponential back off algorithm. It denotes the window size that is last known to be error free. An exemplary code listing for the slow start algorithm is set forth herein in Table 1.

TABLE 1

SAMPLE CODE TO CALCULATE CONGESTION WINDOW:

```
void UpdateCongestionWindo ( Sint32 bytesIncreased )  
{
```

```

// update the congestion window by the number of bytes just
acknowledged.
    if ( fCongestionWindow >= fSlowStartThreshold )
    {
5         // when we hit the slow start threshold, only increase the
           // window for each window full of acks.
           fSlowStartByteCount += bytesIncreased;

           // have we exceeded one window full yet?
           if (fSlowStartByteCount > fCongestionWindow )
10          {
               fCongestionWindow += kMaximumSegmentSize;
               fSlowStartThreshold += kMaximumSegmentSize;
               fSlowStartByteCount = 0;
           }
15      }
      else
          fCongestionWindow += bytesIncreased;

      if ( fCongestionWindow > fClientWindow )
          fCongestionWindow = fClientWindow;
20  }

```

The timer for retransmit time (RTO) is calculated as a modified version of Karn's algorithm used in TCP. The round trip time for each acknowledged packet 18a, 18b is input into an estimator to produce a smoothed round trip time (SRTT) and a running standard deviation (RTTVAR). The retransmit formula is the sum of 25 the smoothed round trip time and four times the standard deviation, or $RTO = SRTT + 4 * RTTVAR$. In addition, one and one half times the RTO as a sample of the first retransmit of any given packet may be initially used to prevent spurious retransmits of any packets when the actual round trip time is increasing rapidly. furthermore, acknowledgments for packets that have been retransmitted are usually not used for 30 packets that have been retransmitted. A listing of exemplary code useful for this aspect of the present invention is set forth in Table 2.

TABLE 2
SAMPLE CODE FOR MAKING SRTT AND RTTVAR CALCULATIONS

```
void AddToEstimate ( Sint32 rttSampleMsecs )
{
5      /*
          Sint32 delta;

      // initialize avg to cur sample, scaled by 2**3
      if ( fRunningAverageMsecs == 0 )
          fRunningAverageMsecs = rttSampleMsecs * 8;

10     // scale average back to get cur delta from sample
      delta = rttSampleMsecs - fRunningAverageMsecs / 8

      // add 1/8 the delta back to the smooth running average
      //same as : rt avg = rt avg + delta / 8, but scaled
      fRunningAverageMsecs = fRunningAverageMsecs + delta ;
15     if ( delta < 0 )
          delta = -1*delta; // absolute value
      /*
          fRunningMeanDevationMsecs is kept scaled by 4

      so this is the same as
20     fRunningMeanDevationMsecs =
fRunningMeanDevationMsecs + ( |delta| - fRunningMeanDevationMsecs ) /4;

          fRunningMeanDevationMsecs += delta -
fRunningMeanDevationMsecs / 4;
      */
25     fRunningMeanDevationMsecs += delta -
fRunningMeanDevationMsecs / 4;

      fCurRetransmitTimeout = fRunningAverageMsecs / 8 +
fRunningMeanDevationMsecs;

      // rto should not be too low..
30     if ( fCurRetransmitTimeout < kMinRetransmitIntervalMsecs )
          fCurRetrasnsmitTimeout = kMinRetrnsmitIntervalMsecs;

      // or too high..
```

00900T" 06608960

```
if ( fCurRetransmitTimeout > kMaxRetransmitIntervalMsecs )  
    fCurRetransmitTimeout = kMaxRetransmitIntervalMsecs;  
}
```

An inverse to the above described slow start, is back off to quickly slow and
5 find a new equilibrium when a retransmit of any packet occurs. After a retransmit,
SSTH is set to a maximum of one half the current CWND or MSS. After adjusting
the SSTH, CWND is set to the MSS. When a stream begins retransmitting, it will
not readjust the CWND or SSTH because of subsequent retransmits until either a
new packet is sent, or packets are acknowledged. Accordingly, the back off
10 prevents a burst of retransmits from closing the windows to unnecessarily low levels.

The client 14 acknowledges each packet successfully received using an RTCP
app packet. A PTCP app packet useful for practicing the above described methods
and procedures is exemplary set forth in Table 3.

TABLE 3

15 **SENDING ACKNOWLEDGEMENTS -**

The client acknowledges each RTP packet successfully received using the
following RTCP app packet:

	# bytes	descriptions
	-----	-----
20	4	rtcp header
	4	SSRC of receiver
	4	app type ("AAAA')
	2	reserved (set of 0)
	2	seqNum

25 Acknowledgments need not be sent for each packet 18a, 18b received at the
client 14, but may be combined into a coalesced acknowledgment. In one
embodiment a coalesced acknowledgment may be a four bit bitmap following the

sequence number of the first acknowledge packet, such that the coalesced acknowledgment could acknowledge up to 33 packets. A coalesced acknowledgment has the advantage of reducing demand for bandwidth in the client 14 to server 12 direction, and reduce processing load at the server 14.

5 Coalesced acknowledgments will be sent whenever the bitmap is full, the sequence numbers "wrap" (restarting at the first sequence number) or whenever a selected time, such as 200 ms, have elapse since the previous acknowledgment and the client 14 has unacknowledged packets. This keeps the round-trip time from becoming artificially large, which may result in unneeded retransmits or lower
10 throughput. Furthermore, the client 14 should send an acknowledgment whenever a packet is received out of sequence.

As seen in Fig. 4, the packets may be transmitted and received, as seen at 50, 52. At 54 a determination is made if the received packet is the first of a sequence. If yes, its ID is placed into the acknowledgement, as indicated at 56. If no, the
15 packet is identified in a bitmap, as indicated at 58. If an acknowledgement is then to be sent in accordance with one of the above conditions, the acknowledgement is coalesced, as indicated at 60 and sent to the server 12, as indicated at 62.

There has been described herein above exemplary preferred embodiments of an improved connectionless protocol. Those skilled in the art may now make
20 numerous uses of and departures from the above described embodiments without departing from the inventive concepts disclosed therein. Accordingly, the present invention is to be defined solely by the scope of the appended Claims.

The Claims

What is claimed as the invention is:

1. A method for real time transmission of information content between a network server and a network client comprising the steps of:

5 transmitting successive packets of said content from said server to a retransmit module;

assigning at said retransmit module to each of said packets a sequence number and a first timer;

10 transmitting further each of said packets from said retransmit module to said network client;

transmitting from said network client to said retransmit module an acknowledgment for each of said packets received at said network client;

15 retransmitting from said retransmit module any of said packets upon expiration of said first timer assigned thereto prior to an acknowledgment for said any one of said packets being received; and

removing from said retransmit module any of said packets upon an occurrence of a predetermined event prior to an acknowledgement for said any of said packets being received.

2. A method as set forth in Claim 1 further comprising:

20 assigning at said retransmit module to each of said packets a second timer wherein expiration of said second timer is said occurrence of said predetermined event.

00900T 0660990

3. A method as set forth in Claim 1 further comprising removing from said retransmit module any of said packets upon said acknowledgment for said any one of said packets being received prior to expiration of said first timer.

4. A method as set forth in Claim 1 further comprising placing said
5 acknowledgment for differing ones of said packets into a coalesced acknowledgment.

5. A method as set forth in Claim 1 further comprising:

maintaining the bandwidth of said successively transmitted packets to the lesser of a congestion window initially determined to be maximum segment size and a client window size no greater than the size of a UDP socket input buffer at said
10 client.

6. A method as set forth in Claim 5 wherein said congestion window is increased by the size of each packet for which an acknowledgment is received.

7. A method as set forth in Claim 6 wherein said congestion window is increased until said congestion window exceeds a predetermined threshold, and
15 increases thereafter by said maximum segment size for each acknowledgment received.

8. A method as set forth in Claim 7 wherein said threshold is determined by a window size that is last known to be error free in receipt of said successively transmitted packets.

009007" 06609960

9. A method as set forth in Claim 7 wherein said threshold is, upon retransmitting of any of said packets, set to the greater of $\frac{1}{2}$ of the current congestion window size or maximum segment size.

10. A method as set forth in Claim 9 wherein said congestion window is
5 reset to said maximum segment size.

11. A method as set forth in Claim 1 wherein said first timer is determined as a function of round trip time defined as a running average time between transmission of each packet and receipt of an acknowledgment for such packet and a standard deviation of each round trip time.

10 12. A method as set forth in claim 11 wherein said function is the sum of said running average plus four times said standard deviation.

13. A method as set forth in claim 11 wherein said first timer is further initially set at one and one-half times said function.

15 14. A method for acknowledging receipt of packets sent from a network server to a network client comprising steps of:
transmitting successively packets from said server;
receiving at said client several of said packets;
placing into a coalesced acknowledgment an ID of a first one of said several of said packets received at said client;
20 adding to said coalesced acknowledgment a bit map identifying selected other ones of said several of said packets received at said client; and
transmitting to said server said coalesced acknowledgment.

009007" 06608960

15. A method as set forth in Claim 14 further comprising sequentially assigning a sequence number as said ID to each of said successively transmitted packets.

16. A method as set forth in Claim 15 wherein said coalesced
5 acknowledgment is sent upon said sequentially assigned sequence numbers being wrapped.

17. A method as set forth in Claim 16 further comprising sending an acknowledgment for any packet having a sequence number out of sequence with said sequence number of an immediately received one of said packets.

18. A method as set forth in Claim 15 wherein said coalesced
10 acknowledgment is sent upon expiration of a predetermined time from a prior coalesced acknowledgment being sent

19. A method as set forth in Claim 18 wherein said coalesced
15 acknowledgment is sent upon expiration of said predetermined time in the event in the event said client has unacknowledged ones of said packets.

20. A method a set forth in Claim 15 wherein said coalesced acknowledgment is sent when said bitmap is full.

21. A network for real time transmission of information content between a network server and a network client comprising:
20 means for transmitting successive packets of said content from said server to a retransmit module;

009007 06608960

means for assigning at said retransmit module to each of said packets a sequence number and a first timer;

means for transmitting further each of said packets from said retransmit module to said network client;

5 means for transmitting from said network client to said retransmit module an acknowledgment for each of said packets received at said network client;

means for retransmitting from said retransmit module any of said packets upon expiration of said first timer assigned thereto prior to an acknowledgment for said any one of said packets being received; and

10 means for removing from said retransmit module any of said packets upon an occurrence of a predetermined event prior to an acknowledgement for said any of said packets being received.

22. A network as set forth in Claim 21 further comprising means for assigning at said retransmit module to each of said packets a second timer wherein
15 expiration of said second timer is said occurrence.

23. A network as set forth in Claim 21 further comprising means for removing from said retransmit module any of said packets upon said acknowledgment for said any one of said packets being received prior to expiration of said first timer.

20 24. A network as set forth in Claim 21 further comprising means for placing said acknowledgment for differing ones of said packets into a coalesced acknowledgment.

25. A network as set forth in Claim 21 further comprising:

09680990-100600

5 26. A network as set forth in Claim 25 wherein said congestion window
is increased by the size of each packet for which an acknowledgment is received.

28. A network as set forth in Claim 27 wherein said threshold is determined by a window size that is last known to be error free in receipt of said successively transmitted packets.

30. A network as set forth in claim 29 wherein said congestion window is reset to said maximum segment size.

31. A network as set forth in claim 21 wherein said first timer is
20 determined as a function of round trip time defined as a running average time
between transmission of each packet and receipt of an acknowledgment for such
packet and a standard deviation of each round trip time.

32. A network as set forth in claim 31 wherein said function is the sum of said running average plus four times said standard deviation.

33. A network as set forth in claim 31 wherein said first timer is further initially set at one and one-half times said function.

5 34. A network for acknowledging receipt of packets sent from a network server to a network client comprising:
 means for transmitting successively packets from said server;
 means for receiving at said client several of said packets;
 means for placing into a coalesced acknowledgment an ID of a first one of
10 said several of said packets received at said client;
 means for adding to said coalesced acknowledgment a bit map identifying selected other ones of said several of said packets received at said client; and
 means for transmitting to said server said coalesced acknowledgment.

15 35. A network as set forth in Claim 34 further comprising means for sequentially assigning a sequence number as said ID to each of said successively transmitted packets.

 36. A network as set forth in Claim 35 wherein said coalesced acknowledgment is sent upon said sequentially assigned sequence numbers being wrapped.

20 37. A network as set forth in Claim 36 further comprising means for sending an acknowledgment for any packet having a sequence number out of sequence with said sequence number of an immediately received one of said packets.

009007" 06608960

38. A network as set forth in Claim 35 wherein said coalesced acknowledgment is sent upon expiration of a predetermined time from a prior coalesced acknowledgment being sent.

39. A network as set forth in Claim 38 wherein said coalesced acknowledgment is sent upon expiration of said predetermined time in the event in the event said client has unacknowledged ones of said packets.

40. A network as set forth in Claim 35 wherein said coalesced acknowledgment is sent when said bitmap is full.

41. A computer readable medium containing a program which implements a procedure for real time transmission of information content between a network server and a network client comprising:

- transmitting successive packets of said content from said server to a retransmit module;
- assigning at said retransmit module to each of said packets a sequence number and a first timer;
- transmitting further each of said packets from said retransmit module to said network client;
- transmitting from said network client to said retransmit module an acknowledgment for each of said packets received at said network client;
- retransmitting from said retransmit module any of said packets upon expiration of said first timer assigned thereto prior to an acknowledgment for said any one of said packets being received; and
- removing from said retransmit module any of said packets upon an occurrence of a predetermined event prior to an acknowledgement for said any of said packets being received.

5 43. A computer readable medium as set forth in Claim 39 further comprising removing from said retransmit module any of said packets upon said acknowledgment for said any one of said packets being received prior to expiration of said first timer.

45. A computer readable medium as set forth in Claim 41 further comprising:

maintaining the bandwidth of said successively transmitted packets to the lesser of a congestion window initially determined to be maximum segment size and a client window size no greater than the size of a UDP socket input buffer at said client.

47. A computer readable medium as set forth in Claim 46 wherein said congestion window is increased until said congestion window exceeds a

predetermined threshold, and increases thereafter by said maximum segment size for each acknowledgment received.

48. A computer readable medium as set forth in Claim 47 wherein said threshold is determined by a window size that is last known to be error free in receipt of said successively transmitted packets.

49. A computer readable medium as set forth in Claim 47 wherein said threshold is, upon retransmitting of any of said packets, set to the greater of $\frac{1}{2}$ of the current congestion window size or maximum segment size.

50. A computer readable medium as set forth in Claim 49 wherein said congestion window is reset to said maximum segment size.

51. A computer readable medium as set forth in Claim 41 wherein said first timer is determined as a function of round trip time defined as a running average time between transmission of each packet and receipt of an acknowledgment for such packet and a standard deviation of each round trip time.

52. A computer readable medium as set forth in Claim 41 wherein said function is the sum of said running average plus four times said standard deviation.

53. A computer readable medium as set forth in Claim 41 wherein said first timer is further initially set at one and one-half times said function.

54. A computer readable medium for acknowledging receipt of packets sent from a network server to a network client comprising steps of:

transmitting successively packets from said server;
receiving at said client several of said packets;
placing into a coalesced acknowledgment an ID of a first one of said several
of said packets received at said client; and

- 5 adding to said coalesced acknowledgment a bit map identifying selected other
ones of said several of said packets received at said client; and
transmitting to said server said coalesced acknowledgment.

55. A computer readable medium as set forth in Claim 54 further
comprising sequentially assigning a sequence number as said ID to each of said
10 successively transmitted packets.

56. A computer readable medium as set forth in Claim 55 wherein said
coalesced acknowledgment is sent upon said sequentially assigned sequence numbers
being wrapped.

57. A computer readable medium as set forth in Claim 56 further
15 comprising sending an acknowledgment for any packet having a sequence number
out of sequence with said sequence number of an immediately received one of said
packets.

58. A computer readable medium as set forth in Claim 55 wherein said
coalesced acknowledgment is sent upon expiration of a predetermined time from a
20 prior coalesced acknowledgment being sent.

59. A computer readable medium as set forth in Claim 58 wherein said
coalesced acknowledgment is sent upon expiration of said predetermined time in the
event in the event said client has unacknowledged ones of said packets.

009001" 06603960

60. A computer readable medium a set forth in Claim 55 wherein said coalesced acknowledgment is sent when said bitmap is full.

61. A computer network comprising:
a server operative to send successive packets into said network;
5 a network client which receives said packets from said network;
a retransmit module responsive to said packets sent by said server to assign to each of said packets a sequence number and a first timer, said transmit module further transmitting each of said packets into said network, said network client further transmitting to said retransmit module an acknowledgment for each of said
10 packets received at said network client, said retransmit module further retransmitting any of said packets upon expiration of said first timer assigned thereto prior to an acknowledgment for said any one of said packets being received, and said retransmit module further removing any of said packets upon expiration of said second timer assigned thereto.

15 62. A network as set forth in Claim 61 wherein said retransmit module further assigns a second timer to each of said packets wherein expiration of said second timer is said occurrence of said predetermined event.

63. A network as set forth in Claim 61 wherein said retransmit module removes any of said packets upon said acknowledgment for said any one of said
20 packets being received prior to expiration of said first timer.

64. A network as set forth in Claim 61 wherein said client is further adapted to place said acknowledgment for differing ones of said packets into a coalesced acknowledgment.

009001" 06608950

65. A network as set forth in Claim 61 where in said server further maintains a bandwidth of said successively transmitted packets to the lesser of a congestion window initially determined to be maximum segment size and a client window size no greater than the size of a UDP socket input buffer at said client.

5 66. A network as set forth in Claim 65 wherein said congestion window is increased by the size of each packet for which an acknowledgment is received.

67. A network as set forth in Claim 66 wherein said congestion window is increased until said congestion window exceeds a predetermined threshold, and increases thereafter by said maximum segment size for each acknowledgment
10 received.

68. A network as set forth in Claim 67 wherein said threshold is determined by a window size that is last known to be error free in receipt of said successively transmitted packets.

69. A network as set forth in Claim 67 wherein said threshold is, upon
15 retransmitting of any of said packets, set to the greater of $\frac{1}{2}$ of the current congestion window size or maximum segment size.

70. A network as set forth in Claim 69 wherein said congestion window is reset to said maximum segment size.

71. A network as set forth in Claim 61 wherein said first timer is
20 determined as a function of round trip time defined as a running average time between transmission of each packet and receipt of an acknowledgment for such packet and a standard deviation of each round trip time.

0098007" 06608960

72. A network as set forth in Claim 71 wherein said function is the sum of said running average plus four times said standard deviation.

73. A network as set forth in Claim 67 wherein said first timer is further initially set at one and one-half times said function.

5 74. A computer network comprising:
a network server which successively transmits packets into said network;
a network client which receives said packets from said network;
said client being adapted to place a coalesced acknowledgment an ID of a first
one of said several of said packets received at said client, and further adapted to add
10 to said coalesced acknowledgment a bit map identifying selected other ones of said
several of said packets received at said client, and further adapted to transmit to said
server said coalesced acknowledgment.

15 75. A network as set forth in Claim 74 wherein said server sequentially assigns a sequence number as said ID to each of said successively transmitted packets.

76. A network as set forth in Claim 75 wherein said coalesced acknowledgment is sent upon said sequentially assigned sequence numbers being wrapped.

20 77. A network as set forth in Claim 76 further comprising means for sending an acknowledgment for any packet having a sequence number out of sequence with said sequence number of an immediately received one of said packets.

78. A network as set forth in Claim 75 wherein said coalesced acknowledgment is sent upon expiration of a predetermined time from a prior coalesced acknowledgment being sent.

5 79. A network as set forth in Claim 78 wherein said coalesced acknowledgment is sent upon expiration of said predetermined time in the event in the event said client has unacknowledged ones of said packets.

80. A network a set forth in Claim 75 wherein said coalesced acknowledgment is sent when said bitmap is full.

009007" 06608960

Abstract of the Disclosure

Packets transmitted from a server into a computer network are assigned a sequence number, a retransmit time and a time to live. Each packet is retransmitted upon the expiration of the retransmit time if no acknowledgment has been received from a client to which the packet was sent. The packet is removed from a retransmit buffer if the time to live timer expires prior to any acknowledgment being received. Multiple acknowledgments may be combined into a coalesced acknowledgment.

009007" 06608960

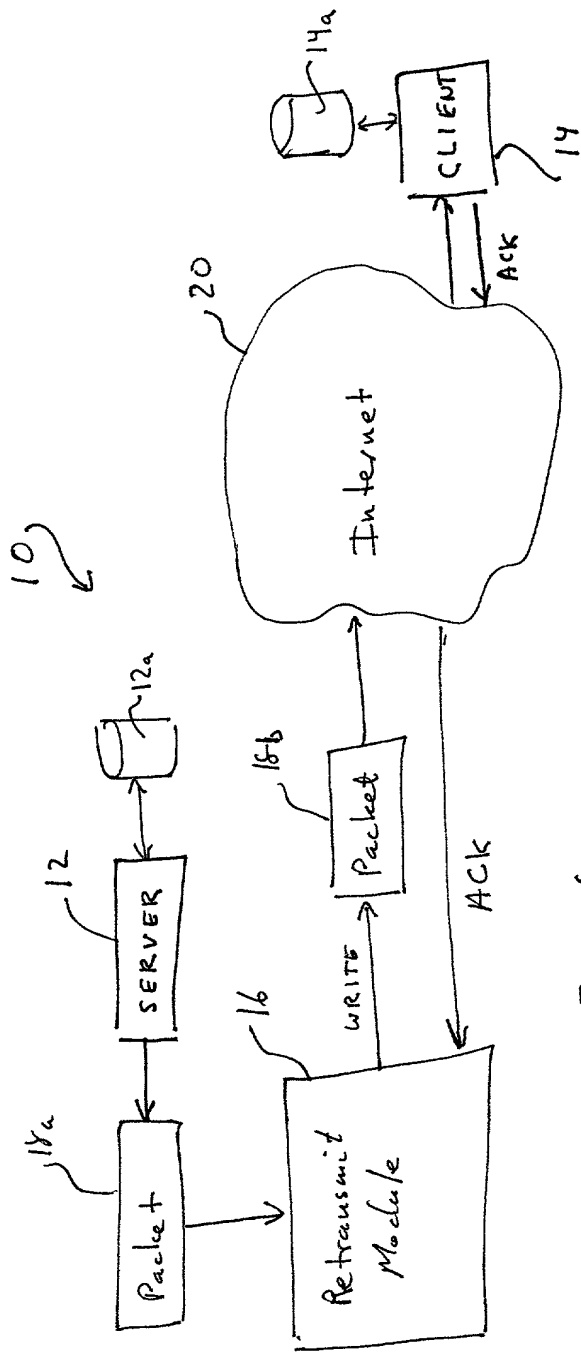


Fig. 1

PACKET	RTU	TTL	SEQ	ACK
A	t_{1a}	t_{2a}	0000	
B	t_{1b}	t_{2b}	0001	
C	t_{1c}	t_{3c}	0010	
:	:	:	:	

Fig. 2

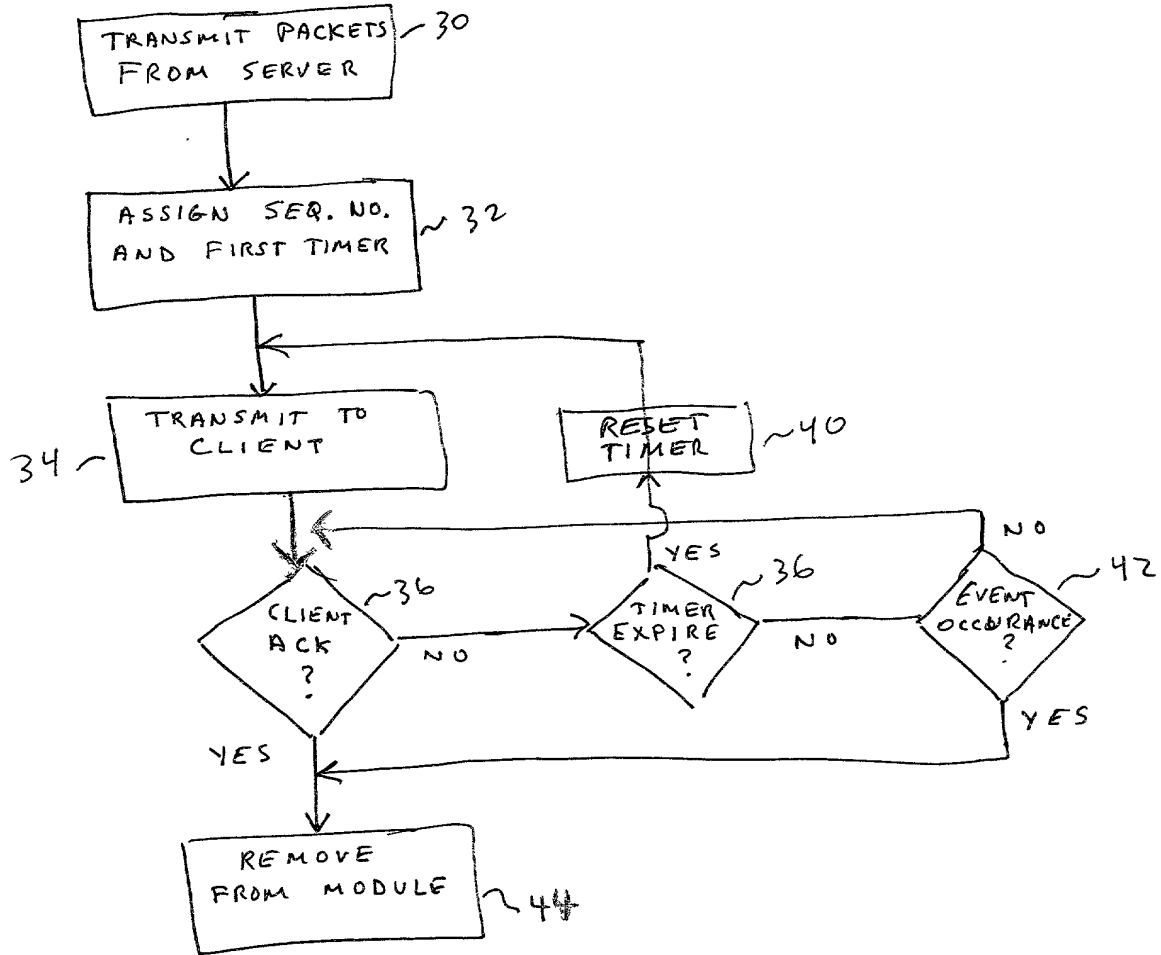


Fig. 3

005001" 06508360

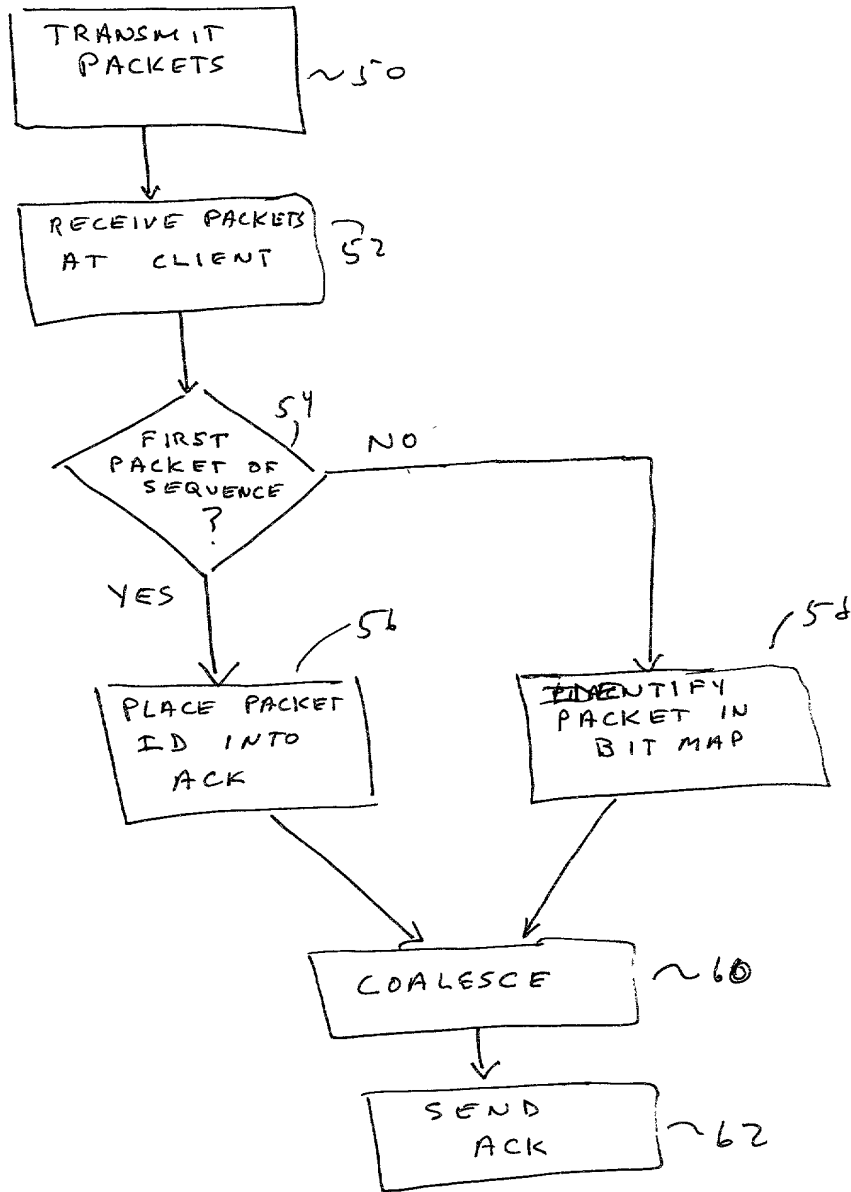


Fig. 4